REMARKS

The specification (abstract) has been objected to in the Office Action. A replacement abstract is hereby provided. No new matter has been added.

Claims 10-14, 17 and 18 have been rejected under 35 USC 102(b) as anticipated by Khoyi. The rejection is respectively traversed.

The invention discloses a circuit arrangement and a corresponding process for data conversion in a processor. A data conversion to be carried out in a processor is carried out in an object-oriented manner before an arithmetic or logical operation, in each case on the basis of a data type specification belonging to the object.

In processors or microprocessors, data type conversion is effected by program sequences before an arithmetic or logical operation with different data types. The data type conversion by program sequences is, however, disadvantage since it reduces the processing speed of a processor or microprocessor. Additionally, this type of data conversion has the disadvantage that the bus system of the processor is loaded by the operation code required for the data conversion. Along with the addressed data type conversion, object-oriented programming languages are used to solve special problems. A data type conversion can also be achieved by an object-oriented command structure. The object-oriented command structure brings with it, however, the disadvantage of an enlargement of the programming code since each command for each combination of data types must be stored in the memory. An enlargement of the programming code also has a reduction of the processing speed as a consequence.

In Khoyi, a process is disclose in which languages can be translated from one language into another (for example, German into English). This system comprises object managers, which recognizes each language respectively (German / English) and translates it into a desired "target" language. The known process has, however, nothing in common with the invention. That is, the fact that processors for translation are also mandatory has nothing to do with the invention.

Two examples (attached hereto) help clarify the differences between the claimed invention and the applied reference:

In the first example, variables are added by a microprocessor with 8086 code (for example, Intel Pentium or AMD Athion) via subprograms, and the result is stored in target variables. All the variables have different types (for example, byte and word). Since the op-code of an 8086 type is dependent (see op-code syntax), type-appropriate subprograms must be called.

In example 2, the same functions are carried out by an object-oriented microprocessor. Since all the source and target variables are identified via their object type (for example, byte and word), the conversion of the object types is carried automatically in the microprocessor and only one sub-program is required.

In Khoyi, an arrangement is disclosed as to how processors are used to solve a problem in the business world. In the present invention, processing or management is described of how, for example, images or presentations can be exchanged between persons from different companies. This conversation can, for example, be done via data processing, such as, for example, PCs, even in different languages such as German or English.

Since the recited structure and method are not disclosed by the applied reference, claims 10 and 17 are patentable. All other claims depend therefrom and are similarly patentable.

Claims 15, 16, 19 and 20 have been rejected under 35 USC 103(a) as unpatentable over Khoyi in view of Microsoft Press Computer Dictionary (Third Edition). The rejection is respectfully traversed for the same reasons presented in the arguments above.

In view of the above, each of the presently pending claims in this application is believed to be in immediate condition for allowance. Accordingly, the Examiner is respectfully requested to withdraw the outstanding rejection of the claims and to pass this application to issue. If it is determined that a telephone conference would expedite the prosecution of this application, the Examiner is invited to telephone the undersigned at the number given below.

In the event the U.S. Patent and Trademark office determines that an extension and/or other relief is required, applicant petitions for any required relief including extensions of time and authorizes the Commissioner to charge the cost of such petitions and/or other fees due in connection with the filing of this document to Deposit Account No. 03-1952 referencing docket no. 449122031600. However, the Commissioner is not authorized to charge the cost of the issue fee to the Deposit Account.

Dated: January 26, 2006

Respectfully submitted,

By _____

Kevin R. Spivak
Registration No.: 43,148
MORRISON & FOERSTER LLP
1650 Tysons Blvd, Suite 300
McLean, Virginia 22102
(703) 760-7762

EXAMPLE 1

Beispiel 1: Code µProzessor 8086

```
Opcode Syntax
88H    mov    m8, r8
89H    mov    m16, r16
8AH    mov    r8, m8
8BH    mov    r16, m16
00H    add    r8, r8
01H    add    r16, r16

ByteSource1   db     12H
ByteSource2   db     23H
ByteResult    db     ?

WordSource1   dw     1230H
WordSource2   dw     3450H
WordResult    dw     ?

       mov al,ByteSource1
       mov bl,ByteSource2
       mov bp,offset ByteResult
       call Byte+ByteToByte

       mov al,ByteSource1
       mov bl,ByteSource2
       mov bp,offset WordResult
       call Byte+ByteToWord

       mov al,ByteSource1
       mov bx,WordSource2
       mov bp,offset WordResult
       call Byte+WordToWord

       mov ax,WordSource1
       mov bx,WordSource2
       mov bp,offset WordResult
       call Word+WordToWord

Byte+ByteToByte:
       add al,bl
       mov byte ptr [bp],al
       ret

Byte+ByteToWord:
       mov ah,0
       mov bh,0
       add ax,bx
       mov word ptr [bp],ax
       ret

Byte+WordToWord:
       mov ah,0
       add ax,bx
       mov word ptr [bp],ax
       ret

Word+WordToWord:
       add ax,bx
       mov word ptr [bp],ax
       ret
```

EXAMPLE 2

Beispiel 2: Objektorientierter µProzessor

```
Opcode Syntax
No1    mov    Reg, TypeData
No2    mov    TypeData, Reg
No3    add    Reg, Reg


ByteSource1   db     12H
ByteSource2   db     23H
ByteResult    db     ?

WordSource1   dw     1230H
WordSource2   dw     3450H
WordResult    dw     ?

       mov Reg1,ByteSource1
       mov Reg2,ByteSource2
       mov Reg3,TypeAddress ByteResult
       call AddToResult

       mov Reg1,ByteSource1
       mov Reg2,ByteSource2
       mov Reg3,TypeAddress WordResult
       call AddToResult

       mov Reg1,ByteSource1
       mov Reg2,WordSource2
       mov Reg3,TypeAddress WordResult
       call AddToResult

       mov Reg1,WordSource1
       mov Reg2,WordSource2
       mov Reg3,TypeAddress WordResult
       call AddToResult

AddToResult:
       add Reg1,Reg2
       mov TypeAddress Reg3,Reg1
       ret
```